ERDC/ITL SR-16-32

Information Technology Laboratory

US Army Corps
of Engineers®
Engineer Research and
Development Center

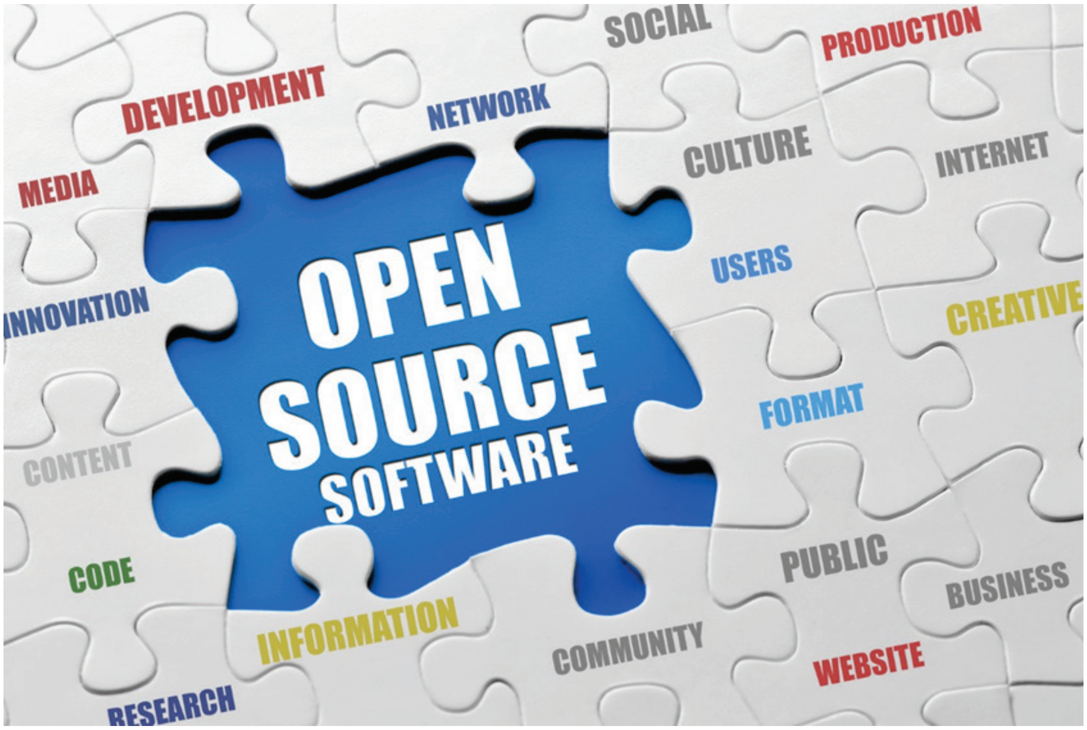
ERDC
INNOVATIVE SOLUTIONS
for a safer, better world

Open Source Software Compliance within the Government

Open Source Software Compliance Policy

Lauren A. Eckert

December 2016



ERDC/ITL SR-16-32
December 2016

Open Source Software Compliance within the Government

Open Source Software Compliance Policy

Lauren A. Eckert

Information Technology Laboratory (ITL)
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Final report

Prepared for U.S. Army Corps of Engineers
Washington, DC 20314-1000

Under Office of Technical Directors (CEERD-IV-T)

Abstract

Open Source Software (OSS) has become increasingly popular for software development and subsequently, government usage has increased. This report outlines a process to manage the risks and complexity of OSS usage within the government. The first step in managing OSS licenses is to understand the requirements regarding compliance, distribution, sharing, attribution, compatibility, termination, copyright, and intellectual property. In order to maintain license compliance, a policy must be created and administered. This policy includes a process of OSS discovery, cataloging, evaluation, review, and approval. Specific guidance is also provided to aid with government acquisitions and contracts as well as information assurance and security compliance requirements. With proper understanding, process implementation, and policy maintenance, the government can effectively use OSS without compliance concerns.

DISCLAIMER: The contents of this report are not to be used for advertising, publication, or promotional purposes. Citation of trade names does not constitute an official endorsement or approval of the use of such commercial products. All product names and trademarks cited are the property of their respective owners. The findings of this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

DESTROY THIS REPORT WHEN NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

Contents

Figures and Tables

Figures

Tables

Preface

This study was conducted for the Office of the Technical Directors.

The work was performed by the Office of the Technical Directors (CEERD-IV-T) of the U.S. Army Engineer Research and Development Center, Information Technology Laboratory (ERDC-ITL). At the time of publication, Dr. Cary Butler, CEERD-IV-T was the Technical Director and Dr. David R. Richards, CEERD-IV-T was the Technical Director. The Deputy Director of ERDC-ITL was Patti S. Duett and the Director was Dr. Reed L. Mosher.

At the time of publication, COL Bryan S. Green was the Commander of ERDC. Dr. Jeffery P. Holland was the Director.

Acronyms

COTS	Commercial Off the Shelf
CIO	Chief Information Officer
DAA	Designated Approval Authority
DFARS	Defense Federal Acquisition Regulation Supplement
DoD	Department of Defense
DoN	Department of the Navy
ERDC	Engineer Research and Development Center
FAR	Federal Acquisition Regulation
FOIA	Freedom of Information Act
FOUO	For Official Use Only
GOTS	Government off the Shelf
GPL	General Public License
IA	Information Assurance
ITL	Information Technology Laboratory
MPL	Mozilla public license
NVDB	National Vulnerabilities Database
OSS	Open Source Software
OSSCP	Open Source Software Compliance Policy
RCERT	Regional Computer Emergency Response Team
TNOSC	Theater Network Operations and Security Center
UTSA	Uniform Trade Secrets Act

1 Introduction

Open Source Software (OSS) has become increasingly popular for software development within the U.S. government. In order to properly use these tools and products, a thorough understanding of the associated licenses as input into the Open Source Software Compliance Policy (OSSCP) is necessary. The overall goal of the OSSCP is to inform the workforce and manage the inherent risks and complexities of OSS without reducing the gains. License and use policies provide a clearer understanding of the software objectives at the business level, and will assist in developing a strategy for use. The OSSCP will also include a governance process, ensuring that the software is being used and maintained correctly. This document outlines OSS policies and the development of the OSSCP that specifically address these needs for government use.

OSS has been defined by the Department of Defense (DoD) as "computer software that includes source code that can be used and modified by the user; this software has been copyrighted and includes a license agreement restriction its use, modification, and distribution" (DoN 2007). It is important to note that open source software is not freeware or public domain software. There are four essential freedoms that make software open source (DoD 2015):

- Free to run the program for any purpose
- Free to study the source code and change it
- Free to share the unmodified software with other people
- Free to distribute modified versions of the software.

The OSSCP requires that the advantages and disadvantages of any software, OSS or proprietary, be carefully assessed within the specific context of intended use on the identified computer network (CENDI 2010). There are several identified advantages of OSS when compared to proprietary software, which can be beneficial in the development of products. The first is that some OSS can be more reliable and secure than other proprietary counterparts because of the public and community-based approach from which it is developed. This type of OSS software typically goes through several reviews that potentially detect defects and vulnerabilities earlier in the design process. Therefore, corrections to the

OSS can be made available earlier within well-established communities. Additionally, due to the nature of OSS, the source code is accessible and allows modifications to be made to fit particular applications as needed. Accessibility to the code also allows for rapid responses if threats are identified and in-depth security reviews and audits if required for certification or other changes. Another benefit is the elimination of contract award risks, to include vendor lock-in. Since OSS can be operated and maintained by multiple suppliers, usage will not result in being locked-in to a specific vendor. Cost savings can also be realized through OSS usage due to the lack of no per-seat or per-copy costs and the lack of maintenance and support costs. Although there are advantages to OSS use over commercial off the shelf (COTS) or government off the shelf (GOTS), the OSSCP should outline how to properly determine if it is the right choice for a given project.

2 Understanding Open Source Compliance

In order to obtain open source compliance, five main steps must be completed (Jacobs and Dawson 2014). These steps will be cyclic in nature and re-occur throughout the fiscal year. In order to implement this process, the below steps must be a central part of the OSSCP.

The first step is to understand the development processes of the organization. During this time the governance committee will be established, with key development professionals in different working areas. This team will then collaborate regarding development, seek to learn and understand policy, and obtain buy in from management. The initial findings of this committee will be incorporated into the policy to ensure it is a good fit for the organization.

The second step involves a full evaluation of all OSS currently in use. This process must identify the type, license, usage, distribution, contributions, and key stakeholders within each project. This will be very detailed understanding that involves all project managers. The identification portion of the process can be aided by available scanning software but should still be reviewed by the project teams. The evaluation will be cyclic in nature and be conducted on an approved schedule to ensure that all software is accounted for and the inventory is up to date.

The third step of the process is to incorporate the reviews into the OSSCP. This establishes the governance committee who will be responsible for all reviews and policy updates. Reviews should result in updates that reflect the development processes, address gaps within the process, and ensure that all components are addressed. Additionally, this policy must gain buy in from all key stakeholders and any concerns identified in the review process should be incorporated. The results of the reviews should be incorporated into the policy at each iteration to ensure that organizational and customer needs are met.

The fourth step of the process is the OSSCP implementation. Key groups must be educated and trained on this policy and OSS guidelines. The key groups would include supervisors, managers, and all project managers at a minimum. This training must be available indefinitely to address any changes in management and projects. Training and implementation will also result in feedback that should be incorporated into this policy.

The fifth and final step of the process is to audit the OSSCP and the processes defined within. At a minimum, annual reviews will be conducted to ensure the policy is accurate and up to date. The policy will be reviewed and updated as necessary during any reorganizations, personnel changes, acquisitions, and other major occurrences.

2.1 License compliance

A major component of the OSSCP is to ensure that key members understand OSS license compliance. The DoD Chief Information Officer (CIO) Memorandum, Open Source Software (OSS) in the Department of Defense (DoD), also known as the Steinbit Memo, states the following:

> DoD Components acquiring, using or developing OSS must ensure that the OSS complies with the same DoD policies that govern Commercial Off the Shelf (COTS) and Government Off the Shelf (GOTS) software. This includes, but is not limited to, the requirements that all information assurance (IA) or IA-enabled IT hardware, firmware and software components or products incorporated into DoD information systems, whether acquired or originated within DoD: comply with the evaluation and validation requirements of National Security Telecommunications and Information System Security Policy Number 11 and; be configured in accordance with DoD-approved security configuration guidelines available at http://iase.disa.mil/ and http://ww:w.nsa.gov/.

This memo defines OSS usage and the requirements that are enforced for use within a government project. (DoD 2003)

2.2 Fundamentals

The OSSCP defines that the OSS and tools will be acquired and used in compliance with all associated licenses. A basic understanding of the licenses, implications, legal disputes, and requirements are crucial to effectively managing OSS, just as they are for managing COTS or GOTS. It is important to remember that DoD has classified OSS as COTS and that compliance with the DoD policies that govern them must be achieved and maintained (DoD 2003). It is also important to note that freeware or shareware, software without a license, provides no permissions to use, copy, or distribute code and should not be confused with OSS (Gruber 2014).

The license management solution, outlined in the OSSCP will contain four main processes (Gruber 2014). The first process will be the license policies and understanding which licenses apply to which use cases as discussed in this section. This process includes identifying and understand each OSS license obligation. The second will be making informed choices and educating developers and project managers on licenses and policies (6 Gruber, 2014). In addition to education on OSS license and policy the developers and managers must understand the specific project and customer requirements. The third step will be the OSS approval process that will be streamlined and automated as much as possible (Gruber 2014). After completion of the third step, the OSS will be in use and the project will be released in a compliant method. The final step is to complete regular auditing to ensure the system is still in compliance, obligations have been met, and no unapproved OSS has been used in the organization (Gruber 2014). By completing this process, with all projects, the risk of OSS can be properly managed.

2.3 Distribution

There are two main types of OSS distribution usage, internal and external, that are key to understanding compliance. For the majority of licenses, software that is not distributed externally, to non-employees, does not require that the source code be shared beyond the organization. However, if software is distributed externally, the licenses should be carefully evaluated to ensure compliance is achieved. Understanding the type, and method, of distribution for each project is key to ensuring compliance is achieved.

DoD has defined software source code and created internal distribution policies that impact OSS. The DoD states that “software source code and associated design documents are “data” as defined by DoD Directive 8320.02” (DoD CIO 2009).

Based on this definition all software source code, regardless of OSS licensing, shall be shared as widely as possible across the DoD to support mission needs (DoD CIO 2009). OSS licenses typically authorize such distributions and therefore allow the OSS to be shared.

The DoD has also defined distribution within the government. The DoD has indicated that there is a misconception regarding distribution and states “many open source licenses permit the user to modify OSS for internal use without being obligated to distribute source code to the public” (DoD CIO 2009).

This extends to an inferred obligation to distribute the source code of any OSS to the public, when used or modified, and therefore it is incompatible with classified or sensitive systems (DoD CIO 2009). Software shared between any members of the U.S. government is considered to be an internal distribution. Additionally, it is considered internal distribution when software is contractually limited to the exclusive use of the government, or running on a third party data center exclusively for government purposes (Molglen et al. 2011) In order for this consideration to be valid, the contract must have a clause specifying the exclusive government use or purpose of the OSS (Molglen et al. 2011). The exception to this rule is the various General Public License (GPLs), which consider all distributions to contractors as outside distribution (Molglen et al. 2011). If the user distributes modified OSS outside of the government, then this is considered external distribution. While these policies define the DoDs stance on distribution requirements, this view has not be reviewed by any court and should be included as part of a risk analysis on any project using OSS (CENDI 2010).

The DoD has also defined when external, or public, distribution of software items, including code fixes and enhancements, should occur based on three conditions (DoD CIO 2009). The first condition is that the responsible government official, a project or program manager, must determine that it is in the government’s best interest to distribute externally. The decision to distribute outside of the government could be made to allow the advantage of future enhancements by others within an open source community. A second condition to be satisfied is that the rights to reproduce and release the software are held by the government. In order to distribute externally the government must also have the rights to authorize others to reproduce and release the software or code item. It is important to remember that when software is developed by government personnel, during duty hours, the government has public release rights. The government would also receive unlimited rights when software is developed by a contractor at the government’s expense or for the government’s exclusive use. The third condition that must be met is that the public release of the software or code is not further restricted by any other law or regulation. An example of such regulations include the Export Administration Regulations and the International Traffic in Arms Regulation, see DoD Directive 5230.24 (i). If all of these conditions are met then an external distribution of governmental software and/or source code can occur.

2.4 Scope of sharing

The scope of sharing is defined by each OSS license and must be understood in order to be compliant with license requirements (Copenhaver et al. 2014). It is important to remember that strongly protective licenses require derivative works and dynamically linked works to be treated as a derived work and distributed under the same license terms as the original (CENDI 2010). This is the main reason that strongly protective licenses are referred to as “viral” since linking, even with proprietary code, forces the release under OSS. Permissive licenses will have less restrictive linking and contribution rules associated and are typically preferable when there are sharing concerns. Table 1, defines the types of sharing allowed by several licenses and is intended as a supplement to understanding the license and requirements. It is also important to note that historically, litigation regarding OSS licenses have resulted in the defendant having to achieve compliance for continued use. It should be understood that payment of damages due to improper sharing compliance may be a portion of compliance litigation, but reaching a compliant state for the software is the main goal of prosecutors in this area. Understanding what constitutes a derived work and the distribution requirements of each license type is key to achieving compliance and understanding the scope of sharing.

2.5 Attribution

The majority of OSS licenses have attribution requirements to acknowledge the authorship of the software (Vasile 2008). These typically require any re-distributor to preserve all copyright notices as defined in the license. Copyright notices can be included within the source code itself, typically at the top of the file, or in the form of a separate file, both of which must be preserved. If copyright notices are not a requirement of the license, then there is typically an attribution to the original developers of the licenses software be made. Attribution in the form of a file, is standard in permissive licenses such as BSD, MIT, and ISC. Understanding the attribution requirement for the software is the only way to ensure compliance.

Table 1. License classifications and basic information.

License	Copy-Left	Permissive	Linking Allowed	Distribution Allowed	Modification Allowed	Patent grant	Private use	Sublicensing	Grants TM
Academic Free License		Yes	Yes		Yes				
Apache License		Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Apple Source License		Yes	Yes		Yes [2]				
Artistic License			Yes [1]		Yes [1]				
BSD License		Yes	Yes	Yes	Yes	Manual	Yes	Yes	Manual
Boost Software License		Yes	Yes		Yes				
Common Development and Distribution License		Yes	Yes		Yes [1]				
Common Public License	Yes		Copy-Left		Copy-Left				
Cryptix General License		Yes	Yes	Yes	Yes	Manual	Yes		Manual
Eclipse Public License			Yes [1]	Limited	Yes [1]	Yes	Yes	Yes [1]	Manual
Educational Community License		Yes	Yes		Yes				
EUPL			Yes [1]		Yes [2]				
GNU Affero General Public License (AGPL)	Yes		Copy-Left[3]	Copy-Left	Copy-Left	Yes	Copy-Left	Copy-Left	Yes
GNU General Public License (GPL)	Yes		Copy-Left[4]	Copy-Left	Copy-Left	Yes	Yes	Copy-Left	Yes
GNU Lesser General Public License (LGPL)	Yes		Yes [1]	Copy-Left	Copy-Left	Yes	Yes	Copy-Left	Yes
IBM Public License	Yes		Copy-Left		Copy-Left				
ISC License		Yes	Yes		Yes				
LaTeX Project Public License		Yes	Yes		Yes				
MIT license / X11 license		Yes	Yes	Yes	Yes	Manual	Yes	Yes	Manual
Mozilla Public License	Yes		Yes	Copy-Left	Copy-Left	Yes	Yes	Copy-Left	No
Netscape Public License			Yes [1]		Yes [1]				
Open Software License	Yes		Yes		Copy-Left				

[1] Yes, with restrictions/limitations [2] Yes with specific list only [3] Copy-Left, only in v3 [4] Copy-Left, only compatible with GPLv

2.6 License fees

Many OSS licenses allow a fee to be charged for providing software. When a redistributor requires a fee for distribution it is typically not a violation of common permissive, or some of the copyleft, licenses. The main difference between the permissive and copyleft licenses is that copyleft license require that once the software is delivered the user has certain freedoms. The BSD and other copyleft licenses do not prohibit charging a fee for a copy of the software. These copyleft licenses typically enforce limits of only charging the cost to physically create the media containing the copy. Additionally, copyright law cannot be used to bypass this copyleft requirement and set those freedoms based on any monetary payment or other condition that would violate the clauses. Also, when obtaining OSS under a copyleft provision, payments cannot be required for continued use of the software. Understanding how licensing fees work within OSS and the specific licenses can benefit the government in cost savings and increase the ability to identify the rights provided under such licenses.

2.7 Compatibility

Compatibility between OSS licenses and proprietary code, are also critical in achieving compliance. Table 2, shows the approval status by various organizations as OSS, the categorization, and the compatibility to create combinatory or combinatory derivative works for several OSS licenses. When identifying OSS compatibility concerns, each license must be considered (Table 2 is meant as reference material to such an identification). When reviewing the Table 2, it is important to understand that there are several OSS organizations, each one having a different stance on compatibility. Therefore, it is important to reference the organization that approves or originated the license in use.

Table 2. This table lists organizations approval status of OSS licenses.

License and version	GPLv3 Compatible	FSF approval	OSI approval	Copyfree approval	Debian approval	Fedora approval
Academic Free License	No	Yes	Yes	No	No	Yes
Apache License 1.x	No	Yes	Yes	No	Yes	Yes
Apache License 2.0	Yes	Yes	Yes	No	Yes	Yes
Apple Public Source License 1.x	No	No	Yes	No	No	No
Apple Public Source License 2.0	No	Yes	Yes	No	No	Yes

License and version	GPLv3 Compatible	FSF approval	OSI approval	Copyfree approval	Debian approval	Fedora approval
Artistic License 1.0	No	No	Yes	No	Yes	No
Artistic License 2.0	Yes	Yes	Yes	No	Yes	Yes
Beerware 42	No	No	No	Yes	No	Yes
Berkeley Database License	Yes	Yes	Yes	No	Yes	Yes
Original BSD license	No	Yes	No	Yes	Yes	Yes
Modified BSD license	Yes	Yes	Yes	Yes	Yes	Yes
Boost Software License	Yes	Yes	Yes	Yes	Yes	Yes
CeCILL	Yes	Yes	Yes	No	Yes	Yes
Common Development and Distribution License	No	Yes	Yes	No	Yes	Yes
Common Public License	No	Yes	Yes	No	Yes	Yes
Creative Commons Zero	Yes	Yes	No	Yes	Partial	Yes
Cryptix General License	Yes	Yes	No	Yes	Yes	Yes
Eclipse Public License	No	Yes	Yes	No	Yes	Yes
Educational Community License	Yes	Yes	Yes	No	No	Yes
Eiffel Forum License 2	Yes	Yes	Yes	No	Yes	Yes
GNU Affero General Public License	Yes	Yes	Yes	No	Yes	Yes
GNU General Public License v2	No	Yes	Yes	No	Yes	Yes
GNU General Public License v3	Yes	Yes	Yes	No	Yes	Yes
GNU Lesser General Public License	Yes	Yes	Yes	No	Yes	Yes
IBM Public License	No	Yes	Yes	No	Yes	Yes
Intel Open Source License	Yes	Yes	Yes	No	No	No
ISC license	Yes	Yes	Yes	Yes	Yes	Yes
LaTeX Project Public License	No	Yes	Yes	No	Yes	Yes
Microsoft Public License	No	Yes	Yes	Yes	No	Yes
Microsoft Reciprocal License	No	Yes	Yes	No	No	Yes

License and version	GPLv3 Compatible	FSF approval	OSI approval	Copyfree approval	Debian approval	Fedora approval
MIT license / X11 license	Yes	Yes	Yes	Yes	Yes	Yes
Mozilla Public License 1.1	No	Yes	Yes	No	Yes	Yes
Mozilla Public License 2.0	Yes	Yes	Yes	No	Yes	Yes
Netscape Public License	No	Yes	No	No	No	Yes
Open Software License	No	Yes	Yes	No	No	Yes
OpenSSL license	No	Yes	No	No	Yes	Yes
PHP License	No	Yes	Yes	No	Yes	Partial
Python Software Foundation License 2.0.1; 2.1.1 and newer	Yes	Yes	Yes	No	Yes	Yes
Q Public License	No	Yes	Yes	No	No	Yes
Reciprocal Public License 1.5	No	No	Yes	No	No	No
Sun Industry Standards Source License	No	Yes	Yes	No	No	Yes
Sun Public License	No	Yes	Yes	No	No	Yes
Sybase Open Watcom Public License	No	No	Yes	No	No	No
W3C Software Notice and License	Yes	Yes	Yes	No	Yes	Yes
XFree86 1.1 License	Yes	Yes	No	No	No	No
zlib/libpng license	Yes	Yes	Yes	No	Yes	Yes
Zope Public License 1.0	No	Yes	No	No	No	Yes
Zope Public License 2.0	Yes	Yes	Yes	No	No	Yes

2.8 Patents

Software can be protected by patent law under certain circumstances. Software protected by a patent is the result of a recent interpretation of the scope of patentable subject matter by courts. The U.S. Patent and Trademark Office began to issue software patents in the late 1990s (Cendi 2010). The software that received patents involved “methods of operation” or “processes” and were covered under business method patents. These business method patents can be difficult and expensive to obtain due to continued controversy. The patent rights in software that is created under a government contract are addressed in FAR 52.227-11 (2014) and 52-227.13 (2014) and DFARS 252.227-7038 (2016).

Patent grants are also different for each license type (Copenhaver et al. 2014). For example; the GPLv3 is licensable, has no terms, allows for owned or controlled, is limited to contribution, expressly excludes infringement due to modification, requires knowledge of reliance on upstream third party licenses, requires that the license extends to all licensees, and provides a discriminatory patent license (Microsoft/Novel deal) (Cendi 2010). The Apache Software License allows for a patent grant when contributing or combining a contribution when the licensed work, is perpetual, licensable and necessarily infringed (Cendi 2010). The Mozilla Public License provides no term, is licensable, and allows for grant on contributions and combinations of contribution and other contributions used by the contributor (Cendi 2010). It is limited if the modifications are only the removal of code or a modification that is absent of relevant contribution. The Apache Software License has no general termination provision, has automatic termination upon filing a patent claim (including cross and counter claims), termination is limited to patent license and is allowable on direct or contributory works (Cendi 2010). Understanding how each specific license allows for patent grants and what they entail is an important part in obtaining compliance in OSS. Additionally, there are some OSS licenses that contain patent licenses. These are usually intended to prevent any patent infringement that would occur through the use of OSS. These licenses prohibit against pursuing patent litigation related to the OSS. Ramifications of these provisions should be well understood since patented code could lose protection if it is merged with OSS (Cendi 2010).

It has been estimated, by the American IP Law Association, that defense against one software patent lawsuit can cost between two and five million dollars (DoD 2016). The End Software Patents coalition reports that 11.4 billion dollars are wasted yearly on patent litigation for software. This figure is based on estimates from 55 software patent suits that were filed each week and includes the $4 million average cost to mitigate a mid-sized patent suit. (DoD 2016)

2.9 Terminations

Each OSS license will contain different termination provisions and understanding what invokes terminations is important (Copenhaver et al. 2014).

The Apache Software License has no general termination provision but will automatically terminate when a patent claim, cross claim, or counterclaim is filed. The termination provisions are limited to patent license of direct or contributory works. The Apache (2004) license states:

> If You institute patent litigation against any entity (including a cross claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

The Mozilla Public License has general termination provisions and termination will occur on patent litigation but excludes any declaratory judgment actions, counterclaims and cross claims. In the event of a termination both copyright and patent will terminate. The Mozilla (2012.) public license states:

> "If you initiate litigation against any entity by asserting a patent infringement claim (excluding declaratory judgment actions, counter-claims, and cross-claims) alleging that a Contributor Version directly or indirectly infringes any patent, then the rights granted to You by any and all Contributors for the Covered Software under Section 2.1 of this License shall terminate."

The GPLv3 also has general termination provisions in which all licenses are terminated. The GPLv3 (2007) license states:

> "You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it."

Understanding the termination clauses, in regard to litigation and other methods, are invoked will allow the users of OSS to ensure they are in compliance even if termination clauses are triggered.

2.10 Copyright and intellectual property

There are additional protections available under copyright law and other types of intellectual property law (CENDI 2010). This includes copyright, trademark, patent, and trade secret law. It is important to note that more than one type of protection can apply to a single piece of computer software.

Computer software is subject to copyright protection under literary works or methods of operation. This is provided under Section 102 of the Copyright Act (1990) and can be protected as “literary works”. The courts and Congress have interpreted “literary works” to include original works of authorship expressed in numbers, words, or other verbal or numerical symbols or indicia which applies to computer software under 17 USC §§ 101 and 102 (a) (1) (CENDI 2010). This copyright protection not only applies to the source code but also extends to the object code. Additionally, “method of operation” refers to the means by which a person operates something and by definition (Section 101 of the Copyright Act), computer programs are sets of statements that create a certain result and therefore a “method of operation” copyright is applicable. The court has not resolved whether “methods of operation” can include a particular expression of expression that may be copyrightable. Methods implemented by software, as “methods of operation,” may also be eligible for patent protection. If a method or work satisfied these requirements it may be copyrightable and protected.

Obtaining copyright on computer software grants specific rights to the owners under Section 106 of the Copyright Act. Under 17 USC § 106 the owners of copyrights to computer software acquire exclusive rights in five ways. These rights include the exclusive right to reproduce and distribute the software. Additionally the right to prepare derivate works based on the original software is exclusive to the owner. Public performance and public display of the software is also an exclusive right granted under the Copyright Act. Each of these five rights provide the ability to control the usage and distribution of the software and can provide a competitive edge to the copyright holder.

Trade secret protection, under state and federal law and various licensing arrangements are also available. The Uniform Trade Secrets Act (UTSA 1985) defines a trade secret as information, including a formula, pattern, compilation, program device, method, technique, or process that meets one of two criteria. The first criteria is that it derives independent economic value, actual or potential, from not being generally known to, and not being readily ascertainable by proper means by, other persons who can obtain economic value from its disclosure or use. The second is that it is the subject of efforts that are reasonable under the circumstances to maintain its secrecy. Protective measures are often employed by computer program trade secret claimants. These measures include licensing agreements with confidentiality provision, non-disclosure agreements for third-party code developers, executable distribution only, and physical security for source code copies. The U.S. Copyright Office provides several registration options for the deposit of just a portion of the code, U.S. Copyright Office Circular No. 61 Registration for Computer Programs, as a recognition of such trade secret claims.

Trademark law can also be applicable in software. This law can be applied to protect source indicators that identify, or form part of, a computer program. These source indicators include names, slogans, designs, graphics, sounds, or other devices that allow a person or entity to identify itself as the source of a computer program. The use of trademark law will allow the owner to protect several aspects of software.

Software is not however considered as an agency record that would be covered by the Freedom of Information Act (FOIA) (CENDI 2010). Computer software that is treated as an agency record typically meets one of three circumstances; 1)It must contain an embedded database that cannot be extracted and is itself releasable under the FOIA, see 32 C.F.R. 518.10 (c), Gilmore v. Department of Energy, 4 F. Supp 2d 912 (N.D. CA 1998), and DeLorme Pub. Co. v. NOAA, 907 F. Supp. 10 (D. Me 1995), 2) that it reveals information about agency policy, functions, decision making or procedures, 3) that it is related to such an accompanying database that the database would be unintelligible or unusable without the software. In some rare instances disclosure may be required and each situation should be reviewed on a case-by-case basis that determines if the data on the software requires it to be treated as such. Releasing software under FOIA will be problematic if it contains sensitive or critical data. Agencies need to carefully consider the security concerns under an open source licensing arrangement where it could be considered an agency record.

2.11 Common licenses

There are several common OSS licenses that have different implications. These licenses are described below along with the specific areas of concern. The Eclipse License, GPL, and MIL are discussed in more detail due to their specific requirements.

2.11.1 Eclipse license

The Eclipse license specifically defines contribution, additions, and linking requirements. This license defines contribution as any subsequent contributor that changes the program or creates additions. Additionally, the contributions made must have been added to the original program by the contributor themselves or by someone acting on their behalf. Contributions do not include additions that are separate modules of software, were distributed with the program under a separate license agreement, and are not derivate works. The license also interprets derivate work in a way consistent with the U.S. Copyright Act and therefore linking may or may not result in a derivate work and is based on the specific circumstances. Based on this information the Eclipse license behaves differently than some of the other OSS licenses and special attention should be paid to these differences.

2.11.2 GPL

The GPL is one of the most prevalent OSS licenses in use. This license type is a copyleft license and enables users to inspect, modify and redistribute software (Moglen et al. 2011). The license states that when a software executable is distributed or conveyed the complete source code must be made available via distribution or an offer for availability. The source must be a complete and an executable version and must be able to be created from the source (meaning that you cannot only distribute the files modified but all required build components). Additionally, the GPL does not require that modified versions of the source code be returned to the original developer or re-integrated, this is considered a branch. Modifications are however, typically re-contributed and result in an innovative OSS ecosystem in which several contributors work together to make the software more robust.

The GPLv2 attempts to control the distribution of derivate and collective works based on software covered by this license. If you distribute or publish any work that contains or is derived from a GPLv2 licensed program, or any part thereof; it must be licensed as a whole at no charge to all third parties under the terms of this license. Identifiable sections of a project that are not derived from the original licensed program, can be considered independent and separate works on their own, and are distributed separately from the licensed software do not have the original license applied to the separate distribution. If you distribute these same sections, which are independent, as a part of the whole based on the licensed program then the distribution must meet all requirements of the license. Based on the fact that the GPL does allow serial and private modifications, or branches, to source code without redistribution to the public the distribution terms are only applicable when the executable is distributed to the public, or "outside" the government.

The GPL license provides rights that resemble Defense Federal Acquisition Regulation Supplement (DFARS) unlimited rights and can also be applied to software acquired under DFARS government purpose rights license. For government purpose rights to be converted to unlimited rights, the distribution of executable and source code must be made internal to the government. This is due to limitations within the DFARS government purpose data rights license. After five–years from the contract award date, government purpose rights will convert to unlimited rights in most cases. Distribution can be made externally if the government purpose software distribution is accompanied by a nondisclosure agreement and the government has unlimited rights. Intellectual property attorneys should always be consulted when questions arise as to the appropriate distribution type.

Government distributions of GPL software can also be subject to other legal limitations (Moglen et al. 2011). These limitations include classification levels, ITAR, distribution statements and export control. Any development using GPL within a classified program is considered a private modification and the executable and source code can only be distributed to individuals with the appropriate clearance. Additionally, within classified programs security law established additional obligations exist. Section 7 of the GPLv2 and section 12 of the GPLv3 state; that if a covered work cannot be distributed or conveyed as to simultaneously satisfy GPL obligations and outside, or security, requirements then the modified software may not be distributed or conveyed. These sections allow GPL software to be developed and modified in classified programs serially and not trigger the requirement to distribute software outside of the classified program, satisfying the GPL obligations and security laws. Since intra-governmental distribution has been classified by the DoD to not be "publication" under U.S. copyright laws, it cannot be considered "conveying" under GPLv3 terms.

Due to the additional legal limitations certain rules are required when GPL executables are distributed by the government. Such executables and source code can only be offered by an authorized delivering entity to an authorized receiving entity. This means that contractors, with authorization, can receive classified source code from the government program office to perform modifications. These modifications are considered an exercise of the freedom to privately modify under the terms of the GPL. These private modifications cannot result in a redistribution to the public and are solely distributed internally to the government.

2.11.2 Mozilla public license (MPL)

The MPL 2.0 defines covered software and modifications. Covered software is defined as the source code from the initial contributor with the source code form license notice attached, the executable form of the source code, and modifications of the source code form. Modifications are defined in two separate ways; the first definition applies to any file in source code form that results from an addition, deletion, or modification of the contents of software covered by the license, the second definition of modification is any new file in source code form that contains any software covered by the license. Even though these definitions sound similar to the GPL, this license does allow for linking. Understanding the implications of each license is the only way to ensure compliance is achieved.

3 Program Administration and Management

This document defines the administration, policy, and management of the OSSCP policy for a DoD organization. This policy establishes a governance committee that is responsible for maintaining this document and enforcing compliance. The governance committee is responsible for the overall approval of OSSCP and for maintaining a list of approved and disapproved OSS that is being used within the program. Each project or program manager will be assigned an OSSCP component owner, also having several roles and responsibilities. Component owners will be authorized to make approval decisions on OSS usage within their projects and will be subject to audits and reviews by the governance committee. The OSSCP will be normally be for official use only (FOUO) and only shared internally within project teams. The policy will be reviewed and updated annually, or when a critical change is identified, by the governance committee. Initial training will be made available for all project managers and supervisors within the organization. After initial training is complete for these target groups, it will be offered yearly for all interested employees. This yearly offering will also allow the new project managers or supervisors to participate. Through careful administration, by a governance committee, the OSSCP will be implemented and maintained for the life of the project.

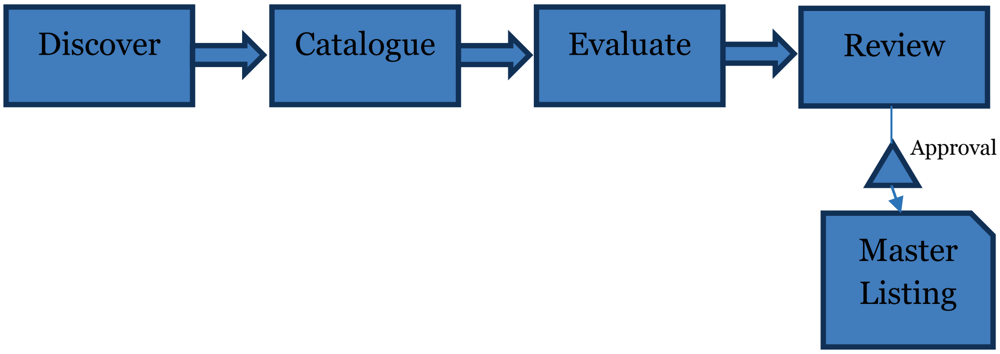
4 OSS Compliance Process

In order to determine the current state of OSS usage within the organization, several steps must be performed (Figure 1).

1. Discovery, during this step a code and build analysis will be conducted for any existing or potential OSS for determination of undocumented usage.
2. Catalogue, during this step all OSS components and tools used within the project are tracked.
3. Evaluate the catalogue, keeping in mind configuration management, version control, proliferation, and the level of support required or provided for the OSS.
4. Review and seek approval for use of all OSS used within the project.

This four step process will result in the creation of a master software listing for each project that should be maintained and enforced by the project manager. At each stage of consideration for OSS inclusion, a full evaluation should be conducted and software on the approved listing should be given preference to any software that has not been reviewed. While each of these steps are targeted as the initial process for determining OSS usage, they must be completed at every major release to ensure that the release is complete and in compliance.

Figure 1. An overview of the four step OSS compliance process.



4.1 Discovery

The discovery process consists of three main steps and should be executed at the project level (Odence and Sheer 2014). For existing projects, the initial discovery process will be very detailed and require a large investment up front. Once the initial discovery is completed, the process will be easier to implement and compliance maintained. For a new project with no existing source code, this first step is not required to be performed until the OSS has reached a release state.

The discovery process is intended to identify all of the OSS currently in use within a project. This in-depth analysis should include examining the software's runtime behavior, searching binary packages as raw data streams, and comparing media distributed with the software (Vasile 2008). During each of these processes the reviewer should be looking for elements of copied OSS and entire OSS packages within the software. To determine if there is any copied OSS element, it must be demonstrated that the copied code was authored by the OSS developer and it was common to both the OSS and the software being evaluated. Specific identification techniques are discussed in section 4.1.1.

During this initial step, additional roles and responsibilities should be assigned to team members as needed. The project or program manager can assign these additional roles and responsibilities in order to ensure all compliance processes are completed. These roles should be assigned early in the process to ensure that the team member is aware of their responsibilities. It is important to note that regardless of delegation, all employees that are assigned functions within this policy should be held responsible for the completion and compliance of those functions.

4.1.1 Identifying OSS through unique identifiers

There are several methods available to aid in the identification of OSS within a software project. If redistribution has occurred, it could obscure OSS origins and initial observations may not be enough to fully identify all usage within the project. Utilizing several different methods is the best way to ensure that critical identification occurs.

One method used to evaluate for OSS usage is to look for unique identifiers. This process attempts to find common elements between an OSS package and the project, specifically looking for copied elements. There are three main ways this evaluation can be completed; examining the program's behavior as it is running, searching the raw data stream of a binary package, or comparing the media distributed with the project distribution. The use of copied OSS is demonstrated when elements are identified that demonstrate authorship and are common in the project and the OSS package. Any elements that are directly comparable or contain unique identifiers, are copyrightable under the OSS package requirements. These items can be as obvious as a digital watermark in graphic content, as simple as a string in code, or as demonstrative as Easter eggs. Additionally, when developing a program it is useful to include these types of unique identifiers to help identify unapproved usage of the custom code created for the project.

The use of unique strings from an OSS package is another sign of usage. Searching the code base for these unique strings (from an OSS distribution) can be a fast way to identify usage. The strings can be code symbols, attributive marks, documentation, prompts, variable and function names, or anything else that includes non-text elements. If the item is unique to the OSS source, and it originates from the OSS author, then it can be used to identify OSS usage. Although these strings may not carry a copyright on their own, they act as a method to identify the copying of the larger, copyrighted material within the program.

Easter Eggs are a hidden feature in software that easily identify the usage of OSS. They are typically intended to entertain and are difficult to access without prior knowledge. Due to their unique and hidden nature, they are great markers of the original authorship and can be difficult to remove if they intentionally obscure the usage. If an Easter egg is used to demonstrate copying it is important to remember that it may be presentable in legal action and caution should be taken regarding the content, especially if this method is used internally to prove ownership.

4.1.2 Digital watermarks

Another method to identify OSS is the use of digital watermarks and fingerprints. These are mathematical transformations applied to digital media data. These transformations do not always affect how media is displayed for end users but do provide a method that indicates authorship. The presence of these media files is not sufficient to demonstrate copying without the ability to demonstrate the original authorship/creation of the files and have a copyrightable interest. Watermarks are typically easier to demonstrate but additional work is required if the OSS value is found in the media portion of the package.

Other methods of identification include the usage of cutting room scraps or identification of known bugs. Cutting room scraps typically include digital photos, videos, or music where the original high resolution pre-edited source is not distributed. The exclusion of these unedited versions is a demonstrative proof of authorship for the owner. Additionally, identified bugs that can be reproduced in a program can demonstrate copying when the same behavior is shown in both. Although these bugs are usually corrected, their presence can identify when usage has occurred. Several other methods exist to identify usage and should be combined to achieve the best results.

4.2 Catalogue

The second step is to create a record of all identified OSS. This record must be complete and maintained throughout the project lifecycle. The record entry should contain all items below at a minimum and must provide detailed answers for further evaluation in the next stages of the process (example provided in Appendix A). Capturing each of these elements will allow for a thorough evaluation in the next step and also provide the reviewers with detailed information necessary.

4.3 Evaluation

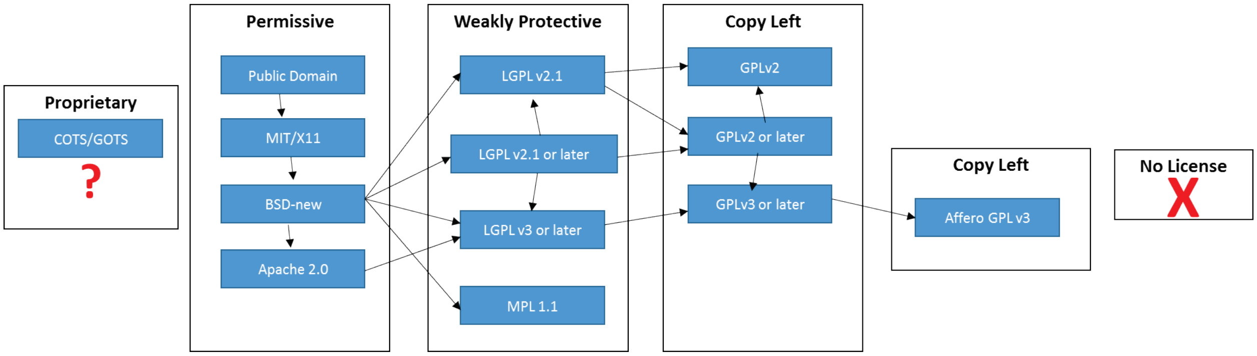
Once the OSS inventory records are complete they must be evaluated to determine if they are appropriate for usage in a project. In order for the governance committee members, project managers, and project team members to perform a thorough evaluation, the following criteria must be addressed:

- Architectural compatibility
- License compatibility
- Modifications needed?
  - Allowed?
- Code:
  - Quality
  - Stability
  - Maturity

- Documentation:
  - Complete?
  - Quality
- Community
  - Active?
  - Mature?
- Commercial support available
  - Support company health
  - Fee associated with support
  - Complete support?
  - Stabile releases offered?
- Security risk
- IP risk

The first such criteria is the architectural compatibility of the OSS with the Olsen 2011). For example, if the OSS license requirements do not allow for inclusion within an executable delivery, then the OSS usage should be denied and usage discontinued (if it was part of an identification effort). Additionally, some OSS will not be compatible with other existing licenses already used within the project. Some OSS publication requirements eliminate compatibility with proprietary software; therefore, existing and future needs must be evaluated. This incompatibility can delay integration of tools, including contractor developed proprietary software (CENDI 2010). Figure 2 is an example of a reference compatibility matrix; however, it should not be used as a substitution for thorough evaluation. In the slide chart below, an arrow between two licenses indicates that they are typically compatible and the resulting combination will be covered under the license being pointed to. Compatibility compliance is the responsibility of the project manager and the use of the slide chart alone does not justify compatibility.

Figure 2. A slide chart depicting license compatibility.



Ensuring that the OSS license permits modification for internal use (when required) and understanding if there is a requirement to distribute source code to the public upon modification is another critical evaluation. If an end-user chooses to distribute modified OSS outside the government then some OSS licenses (such as GPL) will require distribution of the corresponding source code to any downstream recipient of the software (DoD ESI white paper 2015). Therefore, it is important to understand the OSS license in question and how government modified software will be used and redistributed. Any requirement for modifications to be redistributed or reintegrated into the original project should be carefully reviewed to ensure this does not violate the project's requirements, security, or classifications. In classified and other secure computer systems, or where software is export controlled, any modification distribution provisions must be included as a part of the risk assessment. Another critical component is the quality of the OSS code itself along with the stability and maturity. The code quality and stability both lend to more viable OSS that has been through several releases to address any bugs or defects within the code base. Part of the quality evaluation will be the completeness of the documentation, this is critical for usage, configuration, and maintenance of the OSS. The maturity of the OSS and its originating community also lend to more stable products with a higher likelihood of continued maintenance. Additionally, selection of an OSS that does not have an active community that is ready to respond to issues and security vulnerabilities might be a poor choice for production level development. The government might need to invest additional time and effort into the maintenance of an abandoned OSS product and therefore it would become cost prohibitive to the program. The activity level of the community or the health of the commercial support vendor should be considered in determining if the OSS product will meet the project's mission needs. Some OSS may not have a large enough community to ensure the software will remain available during the foreseeable use period of the project. As such, the ability to allocate resources to provide this support must be considered to justify the additional costs (CENDI 2010). If the OSS is offered in a free and paid version, the paid version should always be chosen. The paid version will provide a stable release and offer support to customers while the free version usually will not have a stability guarantee and therefore, the responsibility for ensuring stability will need to be provided by the government. This leads to an evaluation of the support availability, which includes an examination if paid support is offered and if the OSS has an active community. Finally a security and IP risk evaluation should be completed for the OSS. If the software is found to be insecure, incompatible with the project's security needs, or has IP risks that conflict with the project's interests then it cannot be used.

4.4 Review and approval

Before acquiring or using software, whether it is OSS or not, it must be ensured that the terms of the license are compatible with the intended use, users, and identified network for the specific project. In order to meet this objective the results of the evaluation will be provided to the governance committee or an appointed review board for review and approval of usage. This portion of the policy identifies who is allowed to approve OSS for a given use and how the review process will be conducted. DoD policies exist outlining the requirements for approval of OSS, shareware and freeware that must be observed.

"Use of shareware or freeware is prohibited unless specifically approved through IA personnel and by the Designated Approval Authority (DAA) for a specific operational mission requirement and length of time when no approved product exists. Notify Regional Chief Information Officers (RCIOs) and the supporting Regional Computer Emergency Response Team (RCERT)/ Theater Network Operations and Security Center (TNOSC) of local software use approval" (U.S. Army Regulation 25-2 2009).

"Use of open source software (for example, Red Hat Linux) is permitted when the source code is available for examination of malicious content, applicable configuration implementation guidance is available and implemented, a protection profile is in existence, or a risk and vulnerability assessment has been conducted with mitigation strategies implemented with DAA and CCB approval and documentation in the C&A

package. Notify RCIOs and the supporting RCERT/TNOSC of local software use approval" (U.S. Army Regulation 25-2 2009).

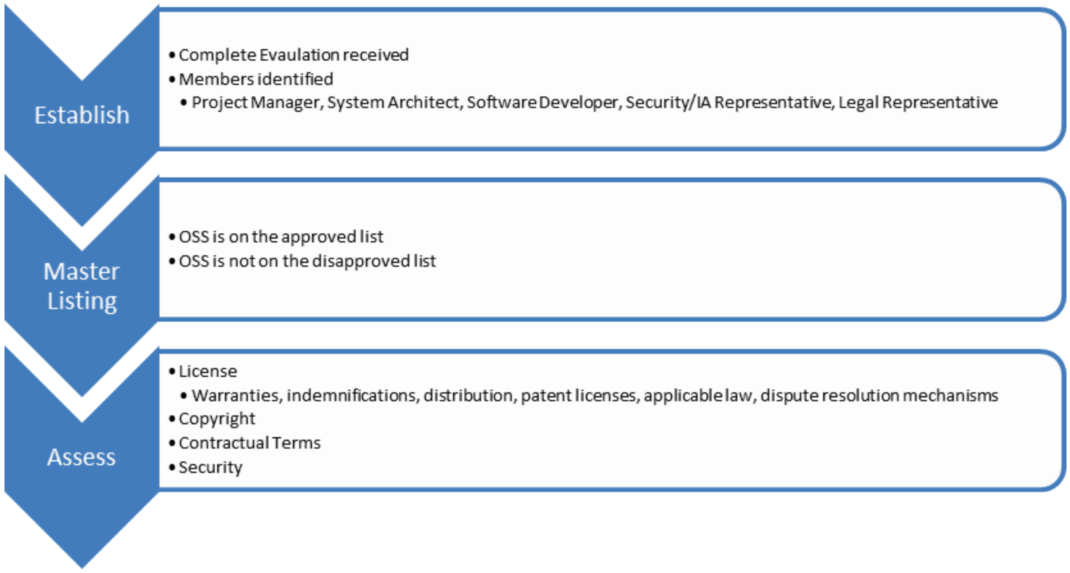
If a review board is established it must consist of experts in software and systems architecture, software development, project management, and a legal representative (Olsen 2011). Since the use of OSS must comply with all lawful licensing requirements, and provisions can be complex, it is encouraged that legal counsel be involved in the review process to fully ensure the implications are fully understood (Olsen 2011) (Figure 3).

Once the full evaluation results are received and the board is convened the first step of the review and approval process can begin. The initial review will consist of identifying the OSS and determining if it is already on the approved or disapproved listing. If the OSS and the requested version is identified in the evaluation is on the approved list, the review process will be simplified to the evaluation of the OSS related to the specifics of the project. If a previous version of the OSS is approved, then the review process must be completed in its entirety. Additionally, any OSS that is on the disapproved list will be reviewed to determine if the specific application of the OSS in the project would allow for an approval and the full process must be completed.

The second step of the review process will assess copyright license and contractual terms, acquisition life cycle, and security. Since the OSS is considered to be a type of proprietary software, the same concerns are of interest in an OSS review prior to purchase or use. All of the contractual terms and copyright licensing requirements should be fully reviewed to ensure the government can legally agree and accept. This includes fully understanding, and accepting, the risks involved. The review should pay particular attention to provisions addressing warranties, indemnifications, distribution and redistribution of code, patent licenses, applicable law, and dispute resolution mechanisms (CENDI 2010).

The review and approval process is designed to be quick and efficient. To achieve this, the burden of proof will be provided during the evaluation process by the project manager. If a project reaches the review and approval process without a sufficient evaluation and inventory record, it will be returned to the project manager with comments and no review will be completed. Additionally, the project manager should be a part of the review and approval process to ensure that there is a quick turn around and project specific expertise is readily available.

Figure 3. Depiction of the review board process and steps.



Establish

- Complete Evaulation received
- Members identified
  - Project Manager, System Architect, Software Developer, Security/IA Representative, Legal Representative

Master Listing

- OSS is on the approved list
- OSS is not on the disapproved list

Assess

- License
  - Warranties, indemnifications, distribution, patent licenses, applicable law, dispute resolution mechanisms
- Copyright
- Contractual Terms
- Security

5 Acquiring OSS

All OSS software has been defined as commercial software in accordance with DFARS 252.227-7014(a) (1) (Michel et al. 2011). Additionally, the OMB addressed *Technology Neutrality* through memo 13, reminding agencies that software competition is important and discrimination is not allowed based on development methods (DoD ESI White Paper 2015). Therefore, there are five main considerations for OSS usage within the government to obtain appropriate statutory preference (DoD CIO 2009).

1. Is the reliability provided through continuous and broad peer-reviews and source code available? This reliability allows for the support of security efforts and the potential for fewer defects.
2. Unrestricted users are another consideration since OSS licenses do not place restrictions on who is allowed to use the software. This offers a net-centric licensing model and enables provisioning for users; whether they are known or unanticipated.
3. Rapid modification is also a consideration since the ability to modify the source code enables rapid response to a change in mission or requirements. This type of software is suitable for experimentation and rapid prototyping since it is provided at minimal or no cost and reduces the burden of creation.
4. An additional consideration relating to cost is that maintenance is shared in OSS and the government can benefit from a reduction in cost of ownership and maintenance compared to GOTS.
5. There is no supplier lock-in where dependence on a particular developer or supplier occurs due to proprietary restrictions. Since OSS can be operated and maintained by several vendors, it reduces any barriers to entry and exits in contracts.

While all of these considerations are relevant they may not be the final decision regarding the use of software. Ultimately, the acquisition and procurement process must choose the software that best meets departmental and mission needs, regardless of OSS status.

5.1 Acquisition and procurement

The acquisition and procurement process contains the most leverage in OSS management (Olsen 2011). There are three major ways that OSS can be procured and used (DoD CIO 2009). The first use is as a component or a single tool within a collection that is used to create an overall product. The second use is as a standalone product where the OSS is marketed to be used on its own (i.e., Alfresco). The third use is as part of a COTS solution that has OSS embedded in the solution but does not market the containment. Each type of procurement and use have different considerations.

Each aspect of the acquisition life cycle should address OSS considerations (CENDI 2010). This includes determining the total cost of ownership associated with any software considered for use or purchase. The low purchase price of OSS is attractive but other costs (ie., updates and fees) may be higher. To mitigate this risk, the characteristics of the software need to be assessed including integrity, reliability, scalability and flexibility. Additional costs that should be considered include transition costs, training costs, and maintenance costs. Transition costs are incurred when software must be configured, installed, backed-up, and conversion or hardware installation are required. Training costs include the training associated with the developers, users, helpdesk staff, and system administrators. The maintenance costs include code tracking, patching, adding functionality, and any onsite maintenance that may be required. Finally the current market share, and growth, of the software should be considered. If the OSS software does not have continuous public maintenance and upgrades, then internal resources may be required to provide all maintenance in place of the OSS community.

It is also important to understand the specific OSS licenses and the legal requirements they impose. In bid and evaluation, as provided in the DFARS, GPL software will be provided to proposers under the license terms (Michel et al. 2011). This type of distribution is considered to be external and appropriate caution must be exercised. All distribution requirements need to be carefully considered prior to release of the proposal request and safeguards should be utilized (i.e., ITAR, export control, classification, or distribution statements). It is also important to consider that although OSS may not require usage payment, the authors or copyright holders may still retain all rights and not allow reverse engineering, modification, or redistribution (DoD ESI Whitepaper 2015). Understanding all provisions within the specific license is necessary by the procurement and acquisition officers.

Several considerations will be unique to the supplier of the software, even if it is not OSS. Any software considered for use, or inclusion, must require the supplier to report each element embedded in the deliverable, whether the element is OSS or not (Olsen 2011). Additionally, the supplier must indicate whether any OSS has been modified to prevent future legal issues from being acquired in the transaction. The software being considered should undergo a license evaluation including a full understanding of the compliance terms. For any code that will be re-distributed after purchase, a warrantee and indemnification statement should be considered as part of the acquisition agreement. Additionally, code scanning should verify the contents and compliance of the OSS terms in the software being procured. All of these considerations will help to mitigate the risks of unknown OSS inclusion by suppliers to the government.

The logistical process is followed by the compliance process containing six steps (Odence and Sheer 2014).

1. Choose the software that best meets mission needs, security requirements, and departmental guidance, this will be completed during the compliance check.
2. Seek approval for the software through the governance committee as described above in section 4.4.
3. The OSS shall be scanned, inventoried and loaded onto the local network.
4. The OSS will be added to the master listing for the project,
5. and into the listing for the approved/disapproved at the lab level.

This is part of the cataloguing requirement to achieve compliance. Once the process has been completed, and the software is deemed secure and compliant it can then be delivered and the logistical process will be complete.

5.2 Contracts and contractual development

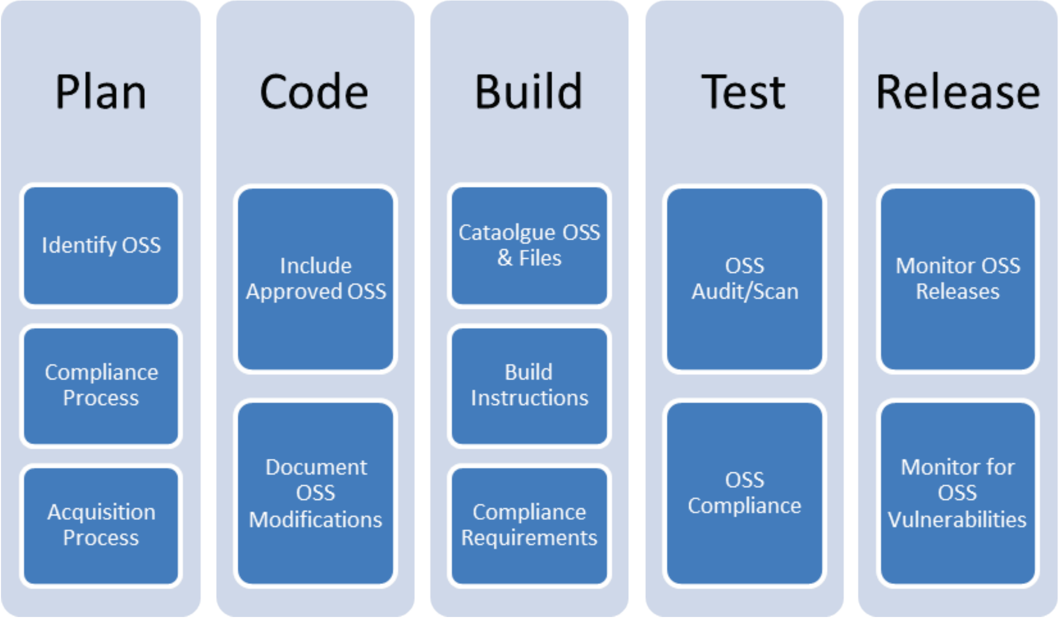
The use of OSS in contractual development is a large area of concern. The government must be aware of contractor usage of OSS when it is embedded or linked to software delivered under a procurement contract, cooperative agreement, or via any other instrument (CENDI 2010). Depending on the OSS used and its associated license, the government may be obligated to provide the source code to the public. Even if contractors use OSS but do not embed it in the delivery, the contract can still require the government to provide source code to the public. All agency procurement officials must add notices in Request for Proposals (RFPs) and contracts regarding the allowance or disallowance of OSS usage, the delivery of identification of any OSS incorporation by the contractor to the government, and a requirement that the contractor provide copies of the OSS version used and respective licenses. These notices are crucial because contractors do not always volunteer information related to OSS code that is licensed, used for development, or delivered to the government under contracts.

As with commercial software code, we must ask the contractor to identify several items in writing. Identification of each type of OSS that is used or modified, including title and version number must be provided. Additionally, each concomitant OSS license and their version numbers must also be provided. Contractors must also identify the asserting party (contractor/sub/awardee) for each OSS item. An indication of whether OSS has been, or will be, modified and by whom must also be provided. Finally, whether modification occurred, or will occur, through incorporating into any third party software should be indicated. While all of this information is required prior to execution of a contract or award, it must also be agreed upon that the full written identification be provided by the contractor and approved by the government before incorporation into a deliverable, use for development of a deliverable, or use to modify or link to preexisting code on a government program or system. Agencies must always request full identification of intended uses and planned OSS that are expected during the performance of a government contract, regardless of whether OSS will be delivered.

6 Code and Documentation Management

An open source governance lifecycle will be adopted to ensure compliance. Open source management works best when it becomes a natural part of the software lifecycle (Olsen 2011). The standard software development lifecycle consists of five main phases that can be modified to address OSS usage as shown in Figure 4 (Gruber 2014).

Figure 4. Depiction of the modified software development lifecycle for OSS.



The first phase of a standard lifecycle is to plan. During the modified version of this phase all potential OSS will be included in the plan and acquired through the compliance and acquisition processes. The second standard phase is to code the solution, which will be dependent on the OSS approval process prior to any inclusion or use. This stage includes tracking and documenting all internal modifications to the OSS in use as part of the catalogue. The third phase is to build the solution, this process will also include updating the catalogue for all OSS in use. Maintenance of the catalogue includes archiving the current OSS source, license, build instructions and files, and any other required files. Additionally, a list of all compliance requirements for distribution will be generated and stored with the archive. Many OSS licenses, including the GPL, allow for required notices or files to be included in a single location. However, some developers prefer to attach a notice to each source file in a project to denote authorship and copyright (SFLC 2007). Either method is acceptable as long as it meets the compliance requirements of the license. The fourth stage is testing of the product, this will now include an audit of all OSS to ensure there is no undocumented software and that license compliance can be achieved (Gruber 2014). The final stage of the lifecycle is the release and this will be updated to involve monitoring of the OSS products used. For each OSS component, vulnerability reports will be monitored as well as bug fixes and the fixes must be shared amongst all applications and users (Olsen 2011).

This policy will also establish a single source, or target of truth for OSS. By establishing a binary repository several benefits can be realized. To create such a repository all OSS source code in use must be procured, all dependencies within the OSS must be identified and also procured, and binary builds will be created. The network location will become the intranet source for all OSS dependencies. This provides a controlled, fast, and highly available repository for OSS used within projects. Additionally it offers a target for deployments that is private, secure and structured. The single repository also provides traceability since all source is in a single location, with dependencies and metadata. The development process will now involve this single source server as a way to ensure compliance.

7 Support and Maintenance

This policy also outlines the requirement for a support plan at the time of OSS component approval (Olsen 2011). When the OSS is approved for inclusion in a software project several responsibilities will be identified. The first will be responsible for tracking security vulnerabilities and bugs. This role is crucial in ensuring that any vulnerabilities or issues are addressed in the OSS and resolved for the final software project. The second will be to notify users, the project team, and the organization of any such vulnerabilities and respective solutions. Typically this role is held by the project manager to ensure that the software project customers are notified as well as appropriate security personnel. Appropriate action must be taken, and a plan established to address the issue, in a timely manner to ensure the government system and hardware are secure and compliant. The plan must include the intended method of resolving the bug or vulnerability. The final item is to evaluate all new releases of the OSS for potential adoption. While this role will be assigned, it is important to proceed through the process of determining if the new release is a candidate for inclusion of the project and all relevant processes must be completed. While these roles can all be held by a single person, the project manager will be responsible for ensuring that all roles are assigned and the requirements are met.

8 Information Assurance and Security

Although the DoD has defined security technical implementation guides (STIGS) through Instruction 8500.2 (2003), *Information Assurance (IA) Implementation*, including Information Assurance Control, “DCPD-1 Public Domain Software controls” it does not forbid the use of OSS. The instructtion limits the use of “binary or machine-executable public domain software or other software products with limited or no warranty.”(DoD CIO 2009).

Since such items can be difficult, or impossible, to review, repair, or extend based on the lack of access to original source code and ownership this limitation was created. Since OSS source code is available to review, repair, or extend, it is not forbidden by this policy. Additionally the DoD has indicated that an information assurance risk exists with any software offered without the appropriate maintenance and support. As such, before approving use of any software, including OSS, the managers and Designated Approving Authorities (DAAs) need to ensure that the support provided is adequate to meet the mission needs. This support can be commercial or provided by a government program office.

A security process must also exist and be enacted (Gruber 2014). This process will involve the verification of the OSS during selection and approval. The National Vulnerabilities Database (NVDB) should be checked during selection for any known security vulnerabilities in the OSS component. If any vulnerabilities are found, the approval process cannot continue. All components that are reviewed will be added to the master list of approved or disapproved software indicating the security review results. Additionally, after the OSS is in use the NVDB must be monitored constantly to ensure all catalogued components do not have new vulnerabilities identified. If at any time, a vulnerability is identified in a catalogued component, the appropriate actions must be taken to mitigate the risks. This includes checking for version updates that may be available to resolve the issue. After an issue is identified or a new version approved for use, the catalogue must be updated to reflect the version number, or solution that is now approved for use or the OSS will be moved to the disapproved listing.

9 Community Participation

One of the benefits of OSS is the community participation to enhance, correct, and extend the code base. There are several levels of community participation with OSS projects, and subject to licensing and they should be observed when possible. These participation levels include:

- None
- Conference presentation
- Bug fix contribution
- Documentation contribution
- Functionality contribution
- Creation of new project

If no community participation is desired and the license permits branching, then the burden of updating and maintaining the branch of the OSS falls solely on government personnel. Any updates to the OSS version made by the community will need to be reviewed to determine if they will be incorporated within the branched version for use. Additionally, projects and programs containing OSS may be part of a presentation and in accordance with the legal and security requirements are allowed. Contributions made by the government, to include bug fixes, documentation, functionality, or new projects must complete the process for releasing OSS and will require organizational attribution as specified by policy.

Two types of contributions are disallowed by this policy. The first, contribution to commercial intermediaries typically provide stable and versioned releases of OSS for a fee. If the government modifies the OSS they cannot return the modified version to the commercial intermediary for incorporation within the OSS version provided. Modifications of this nature should be made by the commercial intermediaries and provided to the government with a limited liability statement or the government must follow the OSS policy for releasing software to the public. The second type of contribution is from a personal account and specifically excludes government attribution. Any works created by an employee, or under a government contract, are owned exclusively by the government. An employee is not permitted to contribute their work under their own account, and without attribution, since the government owns the work and proper release criteria, including classification levels, have not been evaluated or reviewed.

10 Conclusion

The use of an Open Source Software Compliance Policy will greatly benefit the government with regard to OSS usage and compliance. A complete and detailed policy that can inform staff, define processes and procedures, manage risks, and provide benefit will be necessary as we continue to work in a world where OSS is readily used and available. The final policy should define a governance board to implement and aid the workforce in the selection, implementation, and maintenance required for these types of products and tools.

The implementation of such a policy will need to be tiered to address OSS currently in use and the path forward. Organizations will need to define a plan for future use and approval of OSS that meets the requirements and recommendations of the OSSCP. Once the future usage is ensured to be compliant and appropriate the organization can begin to do an inventory of existing OSS and address all compliance issues as they are encountered. These efforts will take a commitment of both time and resources in order to be completed and should be started immediately to avoid any potential violations or repercussions from inappropriate use.

While this technical report provides a background and overview of such an OSSCP, further refinements must be made to adapt an OSSCP and governance board to an organizations specific mission and structure. Once these refinements are made, and official policy has been approved for the organization, the governance board will need to be active to implement the policy at its inception.

References

Apache Software Foundation. January 2004. Apache License Version 2.0. Accessed 10 August 2016. http://www.apache.org/licenses/LICENSE-2.0.html.

CENDI Copyright Working Group. 2010. *Frequently asked questions about copyright and computer software: Issues affecting the U.S. government with special emphasis on open source software*. Ed. Allums V. and Kremers K. CENDI Secretariat: Oak Ridge, TN. October 1, 2010. Accessed on 11 May 2016 http://www.cendi.gov/publications/09-1FAQ_OpenSourceSoftware_FINAL_110109.pdf.

Copenhaver, K., and M. Radcliffe. 2014. *Understanding lesser known elements of open source licenses for use in the connected enterprise*. Lecture, Black Duck Software, Burlington, Massachusetts. http://advance.blackducksoftware.com/content/WRLegalConnectedEnt.

Copyright Law of the United States. Washington, DC: U.S. Copyright Office. Accessed 10 Aug 2016. http://www.copyright.gov/title17/.

Dawson, A. H., and M. Jacobs. 2014. 5 steps to ensuring compliance in the software supply chain: The Harman case study. Lecture, Black Duck Software, Burlington, Massachusetts.

Defense Federal Acquisition Regulation Supplement (DFARS) C.F.R. 252.227-7038 of 2016, Patent Rights – Ownership by the Contractor. Code for Federal Regulations. 2016. Accessed on 10 August 2016. https://www.law.cornell.edu/cfr/text/48/252.227-7038.

Department of Defense (DoD) ESI White Paper. 2015. *Considerations for open source software use: Contractual protections to consider before taking advantage of popular open source solutions*. Department of Defense Chief Information Officer. http://www.esi.mil/contentview.aspx?id=551.

Department of Defense, Chief Information Officer. 2009. "Clarifying Guidance Regarding Open Source Software (OSS)." DoD Memorandum, 16 October 2009. Washington, DC: Department of Defense. Accessed 10 August 2016. http://dodcio.defense.gov/Portals/0/Documents/FOSS/2009OSS.pdf.

Department of Defense. DoD Chief Information Officer. 2009. *Clarifying guidance regarding open source software (OSS)*. Washington, DC: United States Government Printing Office. Accessed on 11 May 2016. http://dodcio.defense.gov/Portals/0/Documents/OSSFAQ/2009OSS.pdf.

Department of the Navy DoN. 2007. *Memorandum from the Chief Information Officer (CIO) Robert J. Carey (5 June 2007)*. Department of the Navy Open Source Software Guidance. Department of the Navy, Washington, DC.

DoD Instruction 8500.2, February 6, 2003 Information Assurance (IA) Implementation. http://www.cac.mil/docs/DoDD-8500.2.pdf.

Federal Acquisition Regulation 48 C.F.R. § 52.227-11 of May 2014, Patent Rights – Ownership by the Contractor. Code for Federal Regulations. https://www.acquisition.gov/far/html/52_227.html.

Federal Acquisition Regulation 48 C.F.R. § 52.227-13 of December 2007, Patent Rights – Ownership by the Government. Code for Federal Regulations. 2007. https://www.acquisition.gov/far/html/52_227.html

Free Software Foundation. 29 June 2007. GNU General Public License GPLv3. Accessed 10 August 2016. https://www.gnu.org/licenses/gpl-3.0.en.html.

Gruber D., and B. Sadogursky, 2014 Managing Open Source and Binaries.

Michel, B., E. Moglen, M. Choudhary, and D. Becker. 2011. Government computer software acquisition and the GNU general public license. Accessed 29 June 2016. http://docplayer.net/8618220-Government-computer-software-acquisition-and-the-gnu-general-public-license.html.

Mozilla Foundation. 2012. Mozilla Public License Version 2.0. Mountain View, CA: Mozilla Foundation. Accessed 10 August 2016. https://www.mozilla.org/en-US/MPL/2.0/.

Odence, P., and D. Sheer. 2014. *Open source software compliance and security*. Lecture, Black Duck Software, Burlington, Massachusetts. https://info.blackducksoftware.com/web-open-source-compliance-security-LP.html.

Olsen, G. 2013. *Four steps to creating an effective open source policy*. Accessed on 10 August 2016. http://www.ossdirectory.com/knowhow/81_Open_Source_Policy_A4EUR.pdf.

Software Freedom Law Center (SFLC). 2007. *Maintaining permissive-licensed files in a GPL-licensed project: Guidelines for developers*. New York: Software Freedom Law Center.

U.S. Army. Headquarters, Department of the Army. Information Assurance. 2009. By U.S. Army. Washington, DC: United States Government Printing Office.

Uniform Trade Secrets Act with 1985 Amendments: http://www.uniformlaws.org/shared/docs/trade%20secrets/utsa_final_85.pdf

Vasile, James. 2008. *Shareware redistribution of FOSS software*. New York: Software Freedom Law Center. https://www.softwarefreedom.org/resources/2008/shareware.pdf.

Appendix A

Table 3. Example record of all identified OSS.

OSS Name: The OSS's registered name or package name	
Version Number: The version that is in use within the project, not necessarily the latest version offered	
OSS License: Reference the specific OSS license that is used for the software	
Local license copy location: Must be a locally owned location that contains a copy of the license created at the time of consideration for inclusion in the project	
License/Attribution file required (Yes/No): Indicate if the license requires additional files to be made available to the end user	
Local license/attribution file copy location: Locally owned location that contains a copy of the file(s) required	
Specific build/release requirements: Any additional constraints in place based on the OSS license	
Licensor: The name of the individual(s)/groups(s) who own the OSS and that the license is applied under	
Business Use: The business functionality that the OSS is being used to achieve, this will be used in the inventory to help others identify evaluated OSS for specific needs	
Will modifications be made (Yes/No): Indication if the OSS is used as is, a no entry, or if changes were/will be made, a yes entry.	
Distribution audience (Internal/External): Indicates the intended distribution for the project the OSS is included or used in	

Distribution method (Website/Stand-alone application/Separate download/Linking): How the OSS is being released to the users. A website release will be common and not require the user to obtain a copy of the software while a stand-alone application would involve sending the users an executable version of the project. The separate download option indicates that the OSS will be offered via download from a separate source and is not required to be incorporated into the release of the project. The linking option indicates that the OSS is being linked to and would be an appropriate choice for any OSS that is included via a library or linked executable where the original source code is not contained within the project itself	
Additional comments: Any relevant information, not already provided, to aid in the evaluation and decision on usage for this OSS	

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
December 2016

2. REPORT TYPE
Final report

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE

Open Source Software Compliance within the Government

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Lauren A. Eckert

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

U.S. Army Engineer Research and Development Center, Information Technology
Laboratory 3909 Halls Ferry Road, Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER

ERDC/ITL SR-16-32

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Corps of Engineers
Washington, DC 20314-1000

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Open Source Software (OSS) has become increasingly popular for software development, and subsequently, government usage has increased. This report outlines a process to manage the risks and complexity of OSS usage within the government. The first step in managing OSS licenses is to understand the requirements regarding compliance, distribution, sharing, attribution, compatibility, termination, copyright, and intellectual property. In order to maintain license compliance, a policy must be created and administered. This policy includes a process of OSS discovery, cataloging, evaluation, review, and approval. Specific guidance is also provided to aid with government acquisitions and contracts as well as information assurance and security compliance requirements. With proper understanding, process implementation, and policy maintenance, the government can effectively use OSS without compliance concerns.

15. SUBJECT TERMS
Open source software--Law and legislation
License agreements
Government purchasing--Law and legislation
Legislation--Compliance costs
Federal government--United States

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT UNCLASSIFIED	**b. ABSTRACT** UNCLASSIFIED	**c. THIS PAGE** UNCLASSIFIED		50	**19b. TELEPHONE NUMBER** *(include area code)*